

Preconditioning the non-relativistic many-fermion problem

Timour Ten^a, Joaquín E. Drut^{b,*}, Timo A. Lähde^c

^a*Department of Physics, University of Illinois, Chicago, IL 60607-7059, USA*

^b*Department of Physics, The Ohio State University, Columbus, OH 43210-1117, USA*

^c*Helsinki Institute of Physics and Department of Applied Physics, Aalto University, FI-02150 Espoo, Finland*

Abstract

Preconditioning is at the core of modern many-fermion Monte Carlo algorithms, such as Hybrid Monte Carlo, where the repeated solution of a linear problem involving an ill-conditioned matrix is needed. We report on a performance comparison of three preconditioning strategies, namely Chebyshev polynomials, strong-coupling approximation and weak-coupling expansion. We use conjugate gradient (CG) on the normal equations as well as stabilized biconjugate gradient (BiCGStab) as solvers and focus on the fermion matrix of the unitary Fermi gas. Our results indicate that BiCGStab is by far the most efficient strategy, both in terms of the number of iterations and matrix-vector operations.

1. Introduction

The Hybrid Monte Carlo (HMC) [1, 2] algorithm provides an efficient way of calculating the properties of many-body Fermi systems. HMC achieves high efficiency by performing global updates by means of Molecular Dynamics (MD) trajectories that are integrated with a finite time step. At each step, the solution of a linear problem

$$M^\dagger M x = b \quad (1)$$

is needed. In this problem, M is a large but typically sparse matrix which provides a real-space representation of the fermion operator that defines the dynamics of the system. While M is typically neither symmetric nor positive definite, it is always possible to consider solving Eq. (1) via conjugate gradient (CG) iteration, as $M^\dagger M$ satisfies these properties [3]. However, as CG is a Krylov-type method based of successive matrix-vector (MV) operations using $M^\dagger M$, the efficiency of CG depends critically on the condition number of that matrix. Unfortunately, in most cases of interest $M^\dagger M$ may be extremely ill-conditioned, leading to a rapid growth of the number of CG iterations required to reach a preset convergence criterion. Such a situation manifests itself in many problems at low enough temperatures or strong enough couplings. Apart from the obvious disadvantages of a rapidly growing computational cost, any iterative method may eventually become unstable if the number of iterations grows without bounds. For these reasons, CG and similar Krylov methods are preferentially used together with a suitable preconditioner. It should be pointed out that while direct methods such as LU decomposition avoid the problems related to ill-conditioning, problems of physical interest are typically large enough such that the storage of M is not feasible. It is thus desirable to resort to “matrix-free” methods based on MV operations only, which require significantly less memory and scale much more favorably with system size.

Whenever M is ill-conditioned, the problem is much more severe for $M^\dagger M$. An attractive option is then to perform the solve the problem in two steps, by considering the linear systems given by

$$M^\dagger y = b, \quad M x = y. \quad (2)$$

*Corresponding author

Email addresses: tten1@uic.edu (Timour Ten), jdrut@mps.ohio-state.edu (Joaquín E. Drut), timo.lahde@tkk.fi (Timo A. Lähde)

However, since M is typically neither symmetric nor positive definite, these equations require more sophisticated algorithms such as biconjugate gradient (BiCG) [4] or stabilized biconjugate gradient (BiCGStab) [5], which is a modern development of the former. One is thus dealing with a much less ill-conditioned problem, at the price of solving two successive problems. Nevertheless, preconditioning remains instrumental in accelerating and stabilizing the solution process. In its simplest form, preconditioning amounts to considering the linear problem

$$PM^\dagger Mx = Pb, \quad (3)$$

where P represents an approximation to $(M^\dagger M)^{-1}$. A similar procedure applies to Eq. (2) given approximations to the inverses of M and M^\dagger . The procedure in Eq. (3) is referred to as “left preconditioning”. So-called “right preconditioning” is also possible by insertion of P between $M^\dagger M$ and x , followed by multiplication with P once the solution has been obtained. In this work, we shall be mainly concerned with the former approach.

While the choice of P is in principle arbitrary, preconditioning is only useful if the solution of Eq. (3) to a given accuracy requires fewer CG iterations than the solution of Eq. (1). Moreover, as iterative methods tend to accumulate roundoff error, an iterative solution may not even be possible without preconditioning. In such cases preconditioning also serves to stabilize the algorithm by enabling convergence. Two obvious extreme choices are those of setting $P = I$ and setting $P = (M^\dagger M)^{-1}$, however no benefits are achieved in either case, as the first one amounts to no preconditioning at all and the second one requires the solution of the original problem in order to determine P . Therefore, what one seeks is compromise between the computational simplicity of the first option and the effectiveness of the second one.

Over the years, many preconditioning strategies were developed and tested on a wide range of problems (see *e.g.* Chapter 11 in Ref. [6], or Chapters 9 and 10 in Ref. [7]). In general, preconditioners specifically designed for a particular problem tend to be superior to generic approaches that build in no knowledge about the physics of the system in question. With this in mind, we will explore several approaches to preconditioning which combine some knowledge of the features of M , M^\dagger and $M^\dagger M$, in order to find a strategy which is both computationally efficient and effective in accelerating the convergence of the linear problem at hand.

In this work, we are mainly concerned with problems that arise in the context of HMC simulations of non-relativistic, (3+1)-dimensional many-fermion systems. The HMC approach forms a central part of state-of-the-art Lattice QCD calculations [8], which involve a relativistic system of fermions. The popularity of HMC stems from the great algorithmic efforts of the Lattice QCD community to enable simulations on large lattices. On the other hand, HMC remains little known in other areas such as condensed-matter physics [9] and nuclear structure [10], where many of the problems are non-relativistic and determinantal Monte Carlo (DMC) [11] remains the method of choice. The most serious drawback of DMC is the poor scaling of the computation time with the lattice volume V , namely $\sim V^3$. In contrast, the use of global updates and iterative solvers enables the HMC algorithm to scale as $\sim V^{5/4}$. In spite of this obvious advantage, the widespread use of HMC implementations for non-relativistic fermions has been hampered by problems related to the operator $M^\dagger M$, which is severely ill-conditioned at low temperatures (see *e.g.* [12]). It is therefore of great interest to explore various preconditioning strategies, as these hold the key to finding a practical HMC implementation which is competitive with current DMC calculations.

The class of problems at hand is characterized by a zero-range interaction, such that the Hamiltonian is given by

$$\hat{H} \equiv \hat{K} + \hat{V}, \quad (4)$$

where the kinetic energy is

$$\hat{K} = \sum_{s=\uparrow,\downarrow} \int d\mathbf{r} \psi_s^\dagger(\mathbf{r}) \frac{-\hbar^2 \nabla^2}{2m} \psi_s(\mathbf{r}), \quad (5)$$

and the potential energy

$$\hat{V} = -g \int d\mathbf{r} \psi_\downarrow^\dagger(\mathbf{r}) \psi_\downarrow(\mathbf{r}) \psi_\uparrow^\dagger(\mathbf{r}) \psi_\uparrow(\mathbf{r}), \quad (6)$$

such that $\psi_s^\dagger(\mathbf{r})$ and $\psi_s(\mathbf{r})$ are creation and annihilation operators for particles of spin s at point \mathbf{r} , respectively. We next seek to recast the grand canonical partition function

$$\mathcal{Z} \equiv \text{Tr} \exp(-\beta(\hat{H} - \mu\hat{N})), \quad (7)$$

in a form amenable to a Monte Carlo calculation. We proceed by discretizing imaginary time into N_τ slices of length $\tau = \beta/N_\tau$ using a Trotter-Suzuki decomposition [13] followed by a continuous Hubbard-Stratonovich transformation (HST) [14] of the form recently proposed by Lee [15]. It should be noted that these last two steps are, to a certain extent, a matter of choice. For instance, continuous-time formulations exist [16] although they have as yet not been combined with HMC. The choice of HST is also dictated by computational preferences [17], although it is not clear at this time whether an HMC implementation is compatible with a discrete HST. The end result is a path integral representation of the partition function

$$\mathcal{Z} = \int \mathcal{D}\sigma (\det M[\sigma])^2 = \int \mathcal{D}\sigma \det M^\dagger[\sigma] M[\sigma], \quad (8)$$

$$M[\sigma] \equiv \begin{pmatrix} 1 & 0 & 0 & 0 & \dots & B_{N_\tau}[\sigma] \\ -B_1[\sigma] & 1 & 0 & 0 & \dots & 0 \\ 0 & -B_2[\sigma] & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & -B_{N_\tau-2}[\sigma] & 1 & 0 \\ 0 & 0 & \dots & 0 & -B_{N_\tau-1}[\sigma] & 1 \end{pmatrix}, \quad (9)$$

where the explicit form of the block matrices

$$B_j[\sigma] \equiv \exp(-\tau K) (1 + A \sin[\sigma_j]), \quad (10)$$

and $A \equiv \sqrt{2\sqrt{\exp(\tau g)} - 1}$, which results from the specific choice of HST. In momentum space, the kinetic energy is given by

$$K_{\mathbf{k},\mathbf{k}'} \equiv \delta_{\mathbf{k},\mathbf{k}'} E(\mathbf{k}), \quad (11)$$

where we have chosen the dispersion relation $E(\mathbf{k}) = \hbar^2 \mathbf{k}^2 / 2m$ for our calculations. The integer vectors $(\mathbf{k}, \mathbf{k}')$ assume values on a three-dimensional momentum lattice given by $k_i = 2\pi n_i / N_x$, where $n_i = -N_x/2, \dots, N_x/2 - 1$. Eq. (9) provides a representation of the fermionic operator M referred to earlier, which is real and manifestly not symmetric.

One of the most popular and successful implementations of HMC is known as the φ -algorithm [2], which combines the stochastic evaluation of the fermion determinant in Eq. (8) with the MD evolution of the auxiliary Hubbard-Stratonovich field σ (originally the gauge field in the case of Lattice QCD). To this end, the pseudofermion representation

$$\det M^\dagger[\sigma] M[\sigma] \equiv \int \mathcal{D}\phi^\dagger \mathcal{D}\phi \exp(-S_p[\sigma]), \quad (12)$$

is applied, giving

$$S_p[\sigma] \equiv \sum_{\mathbf{n}, \tau} [\phi^\dagger Q^{-1}[\sigma] \phi]_{\mathbf{n}, \tau}, \quad (13)$$

for the corresponding action, where $Q \equiv M^\dagger M$. The next step is to introduce an auxiliary field π that will play the role of a conjugate momentum to the field σ in the molecular dynamics evolution. This is accomplished by multiplying the partition function by a constant in the form of a path integral over π ,

$$\mathcal{Z} \equiv \int \mathcal{D}\sigma \mathcal{D}\pi \mathcal{D}\phi^\dagger \mathcal{D}\phi \exp(-\mathcal{H}[\sigma, \pi]), \quad (14)$$

where the MD Hamiltonian is given by

$$\mathcal{H} \equiv \sum_{\mathbf{n}, \tau} \frac{\pi_{\mathbf{n}, \tau}^2}{2} + S_p[\sigma, \pi]. \quad (15)$$

In the pseudofermion formulation, field configurations are sampled by evolving σ and π according to the classical MD equations of motion that follow from the MD Hamiltonian. It should be noted that the pseudofermion field ϕ may be sampled exactly from a Gaussian heat bath and is kept constant during the MD evolution. In practice, the MD evolution should be performed using a reversible symplectic integrator in order to ensure detailed balance. At each step in the MD evolution the “fermion force”

$$F_{\mathbf{n}, \tau}[\sigma, \phi] \equiv \phi^\dagger Q^{-1}[\sigma] \frac{\delta Q}{\delta \sigma(\mathbf{n}, \tau)} Q^{-1}[\sigma] \phi, \quad (16)$$

is calculated, which requires frequent computation of the vector $\eta \equiv Q^{-1}[\sigma]\phi$. This represents the most time-consuming part of the HMC algorithm. One of the most obvious advantages of the φ -algorithm is that the direct calculation of a large determinant is avoided, in favor of the solution of a much smaller linear problem a fixed number of times. Specifically, in our case this involves the repeated solution of Eq. (1) in its preconditioned form given by Eq. (3), where the structure of the matrix is constant but the auxiliary field σ varies at each MD step. This should be contrasted with the cost of computing the full inverse, which is much more expensive than solving a single linear problem. By avoiding the calculation of the determinants and inverses, this algorithm enables global updates of the auxiliary field σ via MD evolution. Moreover, all HMC approaches can be rendered free of systematic errors associated with a finite MD integration step size by means of a Metropolis accept/reject step at the end of each MD trajectory.

In Sec. 2, we study the effect of preconditioning via Chebyshev polynomials, which provide an approximation to Q^{-1} . Chebyshev polynomials provide a generic approach which is easy to apply, but computationally expensive. In Sec. 3, we proceed to study the weak-coupling approximation (WCA) method of preconditioning M and its transpose separately. In Sec. 4 we explore the strong-coupling approximation (SCA) to Q^{-1} which was originally proposed in Ref. [18]. Our findings are summarized in Sec. 5, followed by a concluding discussion in Sec. 6.

2. Chebyshev polynomials

Chebyshev polynomials have frequently been applied in the preconditioning of relativistic quantum field theory problems. In fact, they have led to the development of a whole new class of HMC-type algorithms commonly referred to as Polynomial Hybrid Monte Carlo (PHMC) [20], in which preconditioners are used to separate modes that evolve at different rates in an MD trajectory. The Chebyshev preconditioning technique is based on approximating z^{-1} with a Chebyshev polynomial $P(z)$ of degree $2n$,

$$P(z) \equiv c_{2n} \prod_{k=1}^{2n} (z - z_{2n,k}), \quad (17)$$

where the coefficients are given by

$$c_{2n}^{-1} = \frac{1+\epsilon}{2} \prod_{k=1}^{2n} \left(\frac{1+\epsilon}{2} - z_{2n,k} \right), \quad (18)$$

and the roots by

$$z_{2n,k} = \frac{1+\epsilon}{2} \left[1 - \cos \left(\frac{2\pi k}{2n+1} \right) \right] - i\sqrt{\epsilon} \sin \left(\frac{2\pi k}{2n+1} \right), \quad (19)$$

such that the parameter $0 < \epsilon < 1$ provides a lower bound for the range of validity of the approximation. Indeed, in the interval $[\epsilon, 1]$ the Chebyshev polynomials provide a good approximation, in the sense that the relative error

$$|R_{2n,\epsilon}| \leq 2 \left(\frac{1 - \sqrt{\epsilon}}{1 + \sqrt{\epsilon}} \right)^{2n+1}, \quad (20)$$

where

$$R_{2n,\epsilon}(z) = [P_{2n,\epsilon}(z) - 1/z] z, \quad (21)$$

is exponentially suppressed with the degree of the polynomial.

The preconditioner we seek is obtained from Eq. (17) upon replacing $z \rightarrow M^\dagger M$, assuming that the eigenvalue spectrum of $M^\dagger M$ is normalized in the range $[\epsilon, 1]$. One of the most appealing features of the Chebyshev preconditioning method is that its speed and effectiveness can be tuned by varying the degree of the polynomial n . Moreover, since P is applied in its factorized form, the speed of the preconditioner is directly linked to the speed with which M is applied. The order in which the factors of Eq. (17) are applied turns out to be critical for numerical stability, due to cancellations between the various terms that occur naturally when evaluating a polynomial in finite precision arithmetic. In practice it becomes crucial beyond $n \simeq 20$ to address this issue. Various orderings of the factors were explored in Ref. [19], as well as a recursive Clenshaw algorithm which largely eliminates the accumulation of round-off error. Our experience indicates that the performance of the “bit-reversal” ordering of Ref. [19] is nearly identical to that of the Clenshaw algorithm, and it is slightly faster and requires less memory, in addition to being much simpler to implement. The bit-reversal ordering will therefore be our method of choice.

A serious disadvantage of the Chebyshev approach is that it is only applicable to the preconditioning of the normal equations, as it requires the matrix to have positive definite eigenvalue spectrum. Also, as $M^\dagger M$ becomes extremely ill-conditioned at low temperatures, polynomial orders larger than $n \sim 64$ are needed for effective preconditioning, which makes the Chebyshev approach rather computationally intensive. As will become evident in Sec. 5, this problem can largely be avoided by separately inverting M and M^\dagger , which however requires a different type of preconditioner. We shall now turn to the description of such strategies.

3. Weak coupling expansion

In this section, we demonstrate how the weak-coupling limit of M can be used as the starting point of an efficient preconditioning strategy for M and M^\dagger . It is useful to note that the representation of M given in Eq. (9) can be decomposed as

$$M \equiv M_0 + AM_1 \quad (22)$$

where $M_0 \equiv \lim_{A \rightarrow 0} M$ corresponds to the non-interacting case, and

$$M_1 \equiv \begin{pmatrix} 0 & 0 & 0 & 0 & \dots & C_{N_\tau} \\ -C_1 & 0 & 0 & 0 & \dots & 0 \\ 0 & -C_2 & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & -C_{N_\tau-2} & 0 & 0 \\ 0 & 0 & \dots & 0 & -C_{N_\tau-1} & 0 \end{pmatrix}, \quad (23)$$

where the block matrices are given by

$$C_j \equiv \exp(-\tau K) \sin[\sigma_j]. \quad (24)$$

The inverse M^{-1} can then be formally expanded in powers of AX , giving

$$M^{-1} = (1 + AX)^{-1} M_0^{-1} = (1 - AX + A^2 X^2 - \dots + \dots) M_0^{-1}, \quad (25)$$

where $X \equiv M_0^{-1}M_1$. One can then determine *a posteriori* whether the truncated series represents a good approximation to M^{-1} , which is a reasonable expectation for $A \ll 1$. The most computationally intensive part of such an approximation is the application of M_0^{-1} . However, this matrix is diagonal in frequency-momentum space. It is thus convenient to apply M_0^{-1} after a four-dimensional Fourier transform of the relevant vector, followed by an anti-Fourier transform to return to the original basis. This approach is referred to as Fourier acceleration. In frequency-momentum space, M_0^{-1} is given by

$$[M_0]_{\omega, \mathbf{k}; \omega', \mathbf{k}'}^{-1} \sim \delta_{\omega, \omega'} \delta_{\mathbf{k}, \mathbf{k}'} [1 - \exp(-i\omega - \tau \hbar^2 \mathbf{k}^2 / 2m)]^{-1}, \quad (26)$$

up to a constant normalization factor, with

$$\omega = (2n_\tau + 1)\pi/N_\tau, \quad k_i = 2n_i\pi/N_x, \quad n_\mu = -N_\mu/2, \dots, N_\mu/2 - 1. \quad (27)$$

Care must be taken that the boundary conditions in the temporal direction are antiperiodic, as befit fermions, rather than periodic as commonly assumed in FFT routines. A possible complication arises due to the fact that the radius of convergence of Eq. (25) can be quite limited depending on the spectrum of X , which is unknown for all practical purposes. We therefore turn to techniques which may accelerate the convergence of the expansion.

3.1. Convergence acceleration methods

Here, we will briefly review the practical aspects of the Euler and van Wijngaarden methods for convergence acceleration of series expansions, without concerning ourselves with the underlying theory. The interested reader is referred to the standard literature, see *e.g.* Refs. [21, 22, 23]. Given a sequence of partial sums (in this case of sign-alternating terms)

$$S_{0,k} = \sum_{n=0}^k (-1)^n a_n, \quad (28)$$

where $k = 0, \dots, k_{\max}$, the Euler method consists of defining a set of new sequences given by

$$S_{j+1,k} = pS_{j,k} + (1-p)S_{j,k+1}, \quad (29)$$

where $k = 0, \dots, k_{\max} - (j+1)$. According to this method, instead of using $S_{0,k_{\max}}$ as an estimate of the infinite sum, one can arrive at a much better estimate as follows. Use the original sequence to define new ones using Eq. (29) until all the terms available have been exhausted. The last sum, namely $S_{k_{\max},0}$ will consist of only one term, which is the estimate we seek. In practice one usually takes $p = 1/2$, and it is not uncommon to achieve convergence improvements by several orders of magnitude with a handful of terms as a starting point. It has also been pointed out in the literature that $S_{2k_{\max}/3, k_{\max}/3}$ is often an even better approximation than $S_{k_{\max},0}$. We shall apply this method to the series that results of applying Eq. (25) to a given vector.

The sequence $S_{j,0}$ is commonly referred to as the Euler transform of the original sequence $S_{0,k}$, and it can be shown to be given by

$$S_{j,0} = \sum_{n=0}^j b_n \quad (30)$$

where

$$b_n \equiv \sum_{j=0}^n \binom{n}{j} p^{n-j} (1-p)^{j+1} (-1)^j a_j. \quad (31)$$

The idea of van Wijngaarden was to first multiply each term in the original series by non-vanishing arbitrary constants λ_k and then perform an Euler transformation. It was assumed that a moment generating function $\phi(t)$ exists such that

$$\lambda_k^{-1} = \int_0^\infty \phi(t) t^k dt, \quad (32)$$

and it was shown that the sum of the original series is given by

$$\sum_{n=0}^{\infty} (-1)^n a_n = \sum_{n=0}^{\infty} \mu_n b_n, \quad (33)$$

where

$$\mu_k \equiv \int_0^{\infty} dt \phi(t) \frac{t^k}{(1-p+pt)^{k+1}}. \quad (34)$$

In practice, van Wijngaarden's transformation can turn a slowly convergent or even divergent series into a rapidly convergent series. It can also be shown that the so-called special van Wijngaarden transformation, for which $\lambda_k = s^k/k!$ and $\phi(t) = s \exp(-st)$, where s is a free parameter, does not change the Borel sum of the original series and corresponds to the Laplace transform of the Euler transformation [22].

4. Strong coupling approximation

The idea of constructing a preconditioner based on a strong-coupling approximation was originally developed by Scalettar *et al.* in Ref. [18]. It consists of constructing the inverse of a strong-coupling approximation to $M^\dagger M$, which we shall denote by $\tilde{M}^\dagger \tilde{M}$, where \tilde{M} has the same structure as M but with the substitution

$$B_j \rightarrow B_{0,j} = 1 + A \sin[\sigma_j], \quad (35)$$

where it should be noted that $B_{0,j}$ is diagonal in coordinate space. In this approach, all the blocks will have this property, which makes it extremely inexpensive from a computational point of view. In order to find the inverse, one defines a factorization

$$\tilde{M}^\dagger \tilde{M} \equiv L^\dagger D L, \quad (36)$$

where

$$L^\dagger = \begin{pmatrix} 1 & 0 & 0 & 0 & \dots & 0 \\ -L_1 & 1 & 0 & 0 & \dots & 0 \\ 0 & -L_2 & 1 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & -L_{N_\tau-2} & 1 & 0 \\ L_{N_\tau} & 0 & \dots & 0 & -L_{N_\tau-1} & 1 \end{pmatrix}, \quad (37)$$

and

$$D = \begin{pmatrix} D_1 & 0 & 0 & 0 & \dots & 0 \\ 0 & D_2 & 0 & 0 & \dots & 0 \\ 0 & 0 & D_3 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 0 & D_{N_\tau-1} & 0 \\ 0 & 0 & \dots & 0 & 0 & D_{N_\tau} \end{pmatrix}, \quad (38)$$

for which it is straightforward to show that the individual blocks are given by

$$L_t = D_t^{-1} B_{0,t}, \quad (39)$$

$$D_t = \alpha + B_{0,t}^2 - B_{0,t-1} D_{t-1}^{-1} B_{0,t-1}, \quad (40)$$

for $t = 1, \dots, N_\tau$ and

$$L_{N_\tau} = D_1^{-1} B_{0,N_\tau}, \quad (41)$$

$$D_{N_\tau} = \alpha + B_{0,N_\tau}^2 - B_{0,N_\tau-1} D_{N_\tau-1}^{-1} B_{0,N_\tau-1} - B_{0,N_\tau} D_1^{-1} B_{0,N_\tau}, \quad (42)$$

where the parameter α is tuned such that the number of CG iterations is minimized. The matrix

$$L^{-1}D^{-1}L^{\dagger-1} = (\tilde{M}^{\dagger}\tilde{M})^{-1} \quad (43)$$

then plays the role of the preconditioner in this approach. Notice that the matrices L and L^{\dagger} are trivial to invert, in the sense that the linear problems $Lx = y$ and $L^{\dagger}x = y$ can be solved in a fast and straightforward fashion. For example, $X = L^{\dagger-1}Y$ is given by the recursive expression

$$\begin{aligned} x(r, 1) &= y(r, 1), \\ x(r, 2) &= y(r, 2) + L_1 x(r, 1), \\ x(r, 3) &= y(r, 3) + L_2 x(r, 2), \\ &\dots \\ x(r, N_{\tau}) &= y(r, N_{\tau}) + L_{N_{\tau}-1} x(r, N_{\tau} - 1) - L_{N_{\tau}} x(r, 1). \end{aligned} \quad (44)$$

While Ref. [18] found this approach promising, it presents at least two disadvantages. Firstly, it is not easy to improve in a systematic fashion, in contrast with the Chebyshev approximation. Indeed, one of the key properties of this preconditioner is that it only involves diagonal matrices. Any improvement involving the kinetic energy operator will eliminate this property and make the approach much more computationally expensive. Secondly, just like the Chebyshev method, this preconditioner aims at approximating the inverse of $M^{\dagger}M$, whereas inverting M and M^{\dagger} separately is preferable due to the dramatic increase in the condition number of $M^{\dagger}M$ relative to M . However, it should be pointed out that this preconditioner is computationally very inexpensive and can furthermore be combined with the Chebyshev preconditioner, using the latter as a secondary filter. It is not yet known whether such a combined approach is useful in practice.

5. Results

The number of CG or BiCGStab iterations required to solve the linear problems of Eqs. (1) and (2) is given in Table 1 for the various preconditioners, with a more comprehensive study of the scaling of the second-order weak-coupling preconditioner (WCE2) in Fig. 1. The convergence criterion, referred to as the tolerance parameter, was taken to be a reduction in the norm of the residual by a factor of 10^{-12} . The achieved absolute accuracy is approximately constant for each space-time volume $N_x^3 \times N_{\tau}$, but deteriorates slowly as N_x and N_{τ} are increased. However, it should be noted that in typical HMC simulations, a tolerance parameter of $\sim 10^{-7}$ is considered adequate. Our test runs have been performed using a imaginary time step of $\tau = 0.05$, with couplings of $g = 2.5, 5.0$ and 7.5 . The corresponding values of A are $\simeq 0.52, 0.75$ and 0.95 . The results shown in Tables 1 and 2 correspond to $g = 5.0$, which is close to the unitary limit.

We have used auxiliary field configurations σ with randomly distributed entries in the interval $[-\pi, \pi]$ in order to characterize the performance of the preconditioners. While this is the proper interval for the chosen HST, the distribution differs from that of a thermalized HMC simulation, where the values of σ tend to cluster around $-\pi/2$ and $\pi/2$. However, in practice the net effect of using thermalized instead of random configurations is to increase the number of CG iterations by a factor of ~ 2 across all parameter values. We have confirmed this behavior in a number of cases, and we therefore conclude that the relative merits of the preconditioners remain unaffected by the issue of thermalized versus random configurations. We have also found that 10 random field configurations suffices to provide an estimate of the average number of CG iterations with an uncertainty of less than 10–15% in all cases. The uncertainty can be further suppressed by increasing the number of sample configurations. So far, we have made limited use of Euler–Van Wijngaarden acceleration techniques. For the second-order weak-coupling expansion (WCE2), the simplest Euler method yielded a moderate but definite improvement over the non-accelerated implementation. For the fourth-order expansion (WCE4), the Euler method was found to reduce the number of iterations by up to a factor of 2. Finally, we have varied the right-hand side from a constant vector to one with randomly distributed entries and found no impact on the results.

The number of MV operations consumed by each method is shown in Table 2. Per iteration, CG on the normal equations requires one application of $M^{\dagger}M$ as well as the preconditioner $P(M^{\dagger}M)$. On the

Table 1: Summary of the number of iterations required for the solution of $M^\dagger Mx = b$ with CG, or $M^\dagger y = b$ followed by $Mx = y$ with BiCGStab, for a tolerance of 10^{-12} and a coupling of $g = 5.0$ ($A \simeq 0.75$), which is close to the unitary limit. The columns labeled “*Cheb8*”, “*Cheb16*” and “*Cheb32*” denote, respectively, Chebyshev preconditioning of degrees 8, 16 and 32 using bit-reversal ordering of the roots. The strong-coupling preconditioner is labeled “*SCA*”, and “*WCE0*”, “*WCE1*” and “*WCE2*” denote the weak-coupling expansion of order 0, 1 and 2 in A , respectively. The reported numbers represent averaged over 10 random auxiliary field configurations, which yields an uncertainty below 10-15%. A line denotes the cases that failed to converge in less than 10,000 iterations.

Lattice size $N_x^3 \times N_\tau$	CG				BiCGStab		
	<i>Cheb8</i>	<i>Cheb16</i>	<i>Cheb32</i>	<i>SCA</i>	<i>WCE0</i>	<i>WCE1</i>	<i>WCE2</i>
$6^3 \times 50$	294	167	104	1,790	128	77	55
$8^3 \times 50$	386	226	130	2,388	147	84	62
$10^3 \times 50$	489	259	163	2,810	152	93	67
$6^3 \times 100$	534	316	187	3,470	162	97	71
$8^3 \times 100$	875	460	288	5,223	183	114	78
$10^3 \times 100$	1,102	610	382	7,104	196	118	88
$6^3 \times 200$	864	530	276	5,987	208	119	87
$8^3 \times 200$	3,301	998	604	—	246	148	105
$10^3 \times 200$	—	1,306	735	—	279	162	121

other hand, BiCG and BiCGStab require the application of both M and M^\dagger (as well as their respective preconditioners) for each iteration. Thus, in the absence of a preconditioner, the cost per iteration of CG, BiCG and BiCGStab is the same. The main advantage of using the normal equations with CG is that one avoids the need to successively solve two linear problems. The MV operations performed per CG iteration when using Chebyshev preconditioning can be estimated as follows: the application of a polynomial of degree d to a vector requires $2d$ MV operations; adding the application of the matrix $M^\dagger M$ itself yields a total of $2(d+1)$ operations. Estimating the computational cost of the weak-coupling preconditioner is somewhat more involved. The BiCG or BiCGStab iterations contribute two MV operations from the application of M^\dagger and M . For each power of A , applying K contributes one MV operation to the total cost. However, the non-interacting matrix M_0 is applied using a four-dimensional FFT, which scales as $N_x^3 \times N_\tau \times \log(N_x^3 \times N_\tau)$, which differs from the $N_x^3 \times N_\tau \times \log(N_x^3)$ scaling of the MV operations. The overall computational cost of the approach using the weak-coupling preconditioner of degree d in A is given by $2(d+1)(1+2\beta)$ MV operations, where β is the factor in front of the scaling law for the 4D FFT, relative to that of the 3D FFT. While we have not attempted to estimate β directly (it is implementation-dependent), we find that in practice the gain in CPU time can be lower by a factor of ~ 2 compared with the gain factors quoted in Table 2, where we set $\beta = 1$. Notice however that the gain is still substantial, especially as the total number of MV operations increases only mildly as a function of N_τ .

While the strong-coupling approximation is extremely inexpensive from a computational point of view, its effect on the solution process is rather mild, at least for the values of g considered here. Thus, the number of CG iterations remains rather high, which is an undesirable feature as the iterative solution process is prone to accumulate numerical round-off error after several hundred iterations, which is especially problematic for the BiCG and BiCGStab solvers. As a rule of thumb, in our HMC calculations we aim to keep the iterations below the 200–300 range. Though computationally cheap, a preconditioner which is unable to reduce the iteration count below ~ 1000 is of limited practical usefulness, especially at large N_τ where the number of iterations may increase dramatically. The effective cost of the strong-coupling preconditioner in terms of MV operations is difficult to estimate, as it involves no FFTs. We have found, empirically, that the number of MV operations required per CG iteration can be conservatively estimated to be ~ 3 . In light of the exceedingly high iteration count shown in Table 1 for this preconditioner, however, this MV estimate is irrelevant.

We have also tested the weak-coupling approximation with CG and found that it provides little benefit,

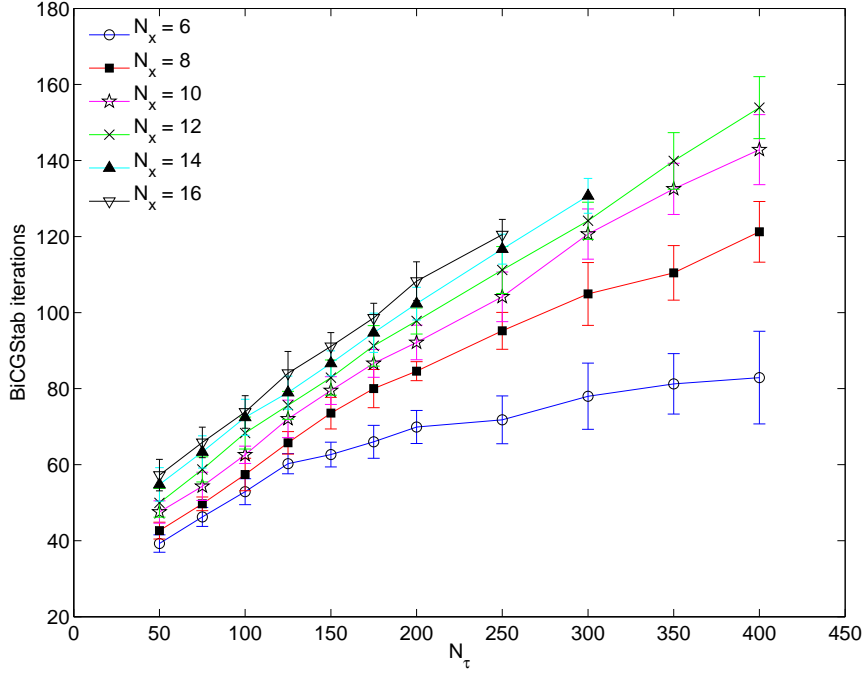


Figure 1: BiCGStab iterations as a function of N_τ and N_x , for the solution of $M^\dagger y = b$ followed by $Mx = y$, for a tolerance of 10^{-7} and a coupling of $g = 5.0$ ($A \simeq 0.75$), using the second-order weak-coupling expansion (*WCE2*) as preconditioner. The results represent averages over 20 random auxiliary field configurations.

in contrast to the dramatic improvement when used with BiCGStab. This situation arises as we have only attempted left preconditioning with CG, and thus the full potential of the preconditioning strategy is not realized. For this to be the case, one would need to precondition with $(M_0^\dagger)^{-1}$ on the left and with M_0^{-1} on the right. In the case of BiCGStab this complication does not appear, as M and M^\dagger are preconditioned separately. Finally, in Table 3 we show the behavior of the weak-coupling preconditioner as a function of the coupling, for $g = 2.5, 5.0$ and 7.5 . We note that the case of $g = 10.0$ ($A \simeq 1.14$), typically failed to converge in less than 15,000 MV operations.

6. Summary and Conclusions

In this work, we have evaluated three different strategies to precondition the non-relativistic many-fermion problem: Chebyshev polynomials, a strong-coupling approximation and a weak-coupling expansion. We have argued that such preconditioning forms a central part of HMC calculations, as the frequent solution of a linear problem involving the ill-conditioned fermion matrix M is at the heart of the HMC algorithm. We have considered the cases of normal equations $M^\dagger Mx = b$ which is tractable using the CG algorithm, as well as the two-step approach of solving $M^\dagger y = b$ followed by $Mx = y$, using the BiCG or BiCGStab algorithms. Our results indicate that both the Chebyshev polynomials (Sec. 2) and the weak-coupling expansion (Sec. 3) can be effective preconditioners, especially when high orders are used. However, from the point of view of performance, Chebyshev polynomials represent an expensive choice. Additionally, the Chebyshev approach requires a matrix with a positive definite eigenvalue spectrum, which forces us to work with the CG solver on the normal equations. This is unfortunate, as M tends to be a rather ill-conditioned matrix (especially at low temperatures), which makes the normal system $Qx = b$ extremely ill-conditioned. The result is that

Table 2: Summary of the number of matrix-vector (MV) operations required for the solution of $M^\dagger Mx = b$ with CG, or $M^\dagger y = b$ followed by $Mx = y$ with BiCGStab. Notation and parameters are as for Table 1. The rightmost column illustrates the gain provided by the lowest-order weak-coupling expansion using BiCGStab over the most favorable case using CG. As explained in the text, the actual gain in CPU time is likely to be somewhat less. Notice in particular the mild increase in the number of MV operations for BiCGStab as a function of N_x and N_τ .

Lattice size $N_x^3 \times N_\tau$	CG				BiCGStab			Gain
	<i>Cheb8</i>	<i>Cheb16</i>	<i>Cheb32</i>	<i>SCA</i>	<i>WCE0</i>	<i>WCE1</i>	<i>WCE2</i>	
$6^3 \times 50$	5,292	5,678	6,864	5,370	768	924	990	7x
$8^3 \times 50$	6,948	7,684	8,580	7,164	882	1,008	1,116	8x
$10^3 \times 50$	8,802	8,806	10,758	8,430	912	1,116	1,206	9x
$6^3 \times 100$	9,612	10,744	12,342	10,410	972	1,164	1,278	10x
$8^3 \times 100$	15,750	15,640	19,008	15,669	1,098	1,368	1,404	14x
$10^3 \times 100$	19,836	20,740	25,212	21,313	1,176	1,416	1,584	17x
$6^3 \times 200$	15,552	18,020	18,216	17,961	1,248	1,428	1,566	12x
$8^3 \times 200$	59,418	33,932	39,864	—	1,476	1,776	1,890	23x
$10^3 \times 200$	—	44,404	48,510	—	1,674	1,944	2,178	27x

Table 3: Effectiveness of the second-order (*WCE2*) and fourth-order (*WCE4*) weak-coupling preconditioners as a function of g . For $g = 2.5$, we have $A \simeq 0.52$, for $g = 5.0$, $A \simeq 0.75$ and for $g = 7.5$, $A \simeq 0.95$. The figures given represent the number of matrix-vector operations required to solve $M^\dagger y = b$ followed by $Mx = y$ with BiCGStab. These are averages over 10 random configurations, with an uncertainty in the range of 10-15%.

Lattice size $N_x^3 \times N_\tau$	$g = 2.5$		$g = 5$		$g = 7.5$	
	<i>WCE2</i>	<i>WCE4</i>	<i>WCE2</i>	<i>WCE4</i>	<i>WCE2</i>	<i>WCE4</i>
$10^3 \times 50$	630	600	1,206	1,320	2,358	4,110
$10^3 \times 100$	702	690	1,584	1,530	3,816	9,330
$10^3 \times 200$	864	780	2,178	2,190	6,444	12,720

the number of MV operations required to solve the problem with a preset accuracy grows rapidly with the size of the problem, in particular with N_τ , which controls the temperature.

We have shown that solving $M^\dagger y = b$ followed by $Mx = y$ using the BiCGStab algorithm provides an alternative, extremely promising approach. The use of a weak-coupling preconditioner supplemented with a simple convergence acceleration technique solves the problem elegantly and provides dramatically enhanced performance in terms of both number of iterations and MV operations, which translates into significant savings of CPU time and improved scaling of the HMC algorithm at low temperature. While we have only provided a first, sketchy comparison of the two methods, the difference between using CG with Chebyshev preconditioning and BiCGStab with weak-coupling preconditioning is so striking that the latter technique is the obvious choice. The weak-coupling preconditioner represents a good example in which knowledge of the structure of the matrix, as well as the physical problem at hand, allows for the construction of an effective and efficient strategy to accelerate the solution process.

The strong-coupling approximation of Ref. [18] provides an alternative to Chebyshev preconditioning. While computationally inexpensive, this preconditioner turned out to be less efficient in terms of its ability to reduce the number of CG iterations. Apparently, this approach fails to capture the relevant physics at the range of couplings in the vicinity of the unitary limit. In an effort to shed more light on this problem, we explored the behavior of the weak-coupling preconditioner as a function of the coupling g , and found that its effectiveness breaks down somewhere between $g = 7.5$ and 10. This situation might be remedied by using a strong-coupling expansion in powers of A^{-1} , although our attempts to apply such an expansion have so far not been successful. However, we believe this to be a topic worthy of further investigation. Alternatively, one

can reduce the value of the imaginary time step τ , which serves to extend the usefulness of the weak-coupling expansion to larger values of g , given that the relevant expansion parameter is $A = \sqrt{2}\sqrt{\exp(\tau g) - 1}$. However, this has the drawback of increasing the value of N_τ required to achieve a given temperature, which can be taxing on the method.

Various cases of interest that could and should be studied lie beyond the scope of this work. Among these are the case of finite effective range, relevant for nuclear and neutron matter calculations, which would make the potential energy operator more dense in real space (it is diagonal for the zero-range interaction considered here). Other dispersion relations, such as that of the conventional Hubbard model should be studied as well, particularly since this might open the door to large-scale HMC simulations in the fields of solid-state and atomic physics. Finally, we have focused here on a problem in $3 + 1$ dimensions, although the methods employed in this study carry over directly to applications in lower dimensions.

Acknowledgements

We would like to thank Dick Furnstahl, Richard Scalettar and Nandini Trivedi for useful discussions and encouragement. We acknowledge support under U.S. DOE Grants No. DE-FG02-00ER41132 and DE-AC02-05CH11231, UNEDF SciDAC Collaboration Grant No. DE-FC02-07ER41457 and NSF Grant No. PHY-0653312. This study was supported in part by the Academy of Finland through its Centers of Excellence Program (2006 - 2011), the Vilho, Yrjö, and Kalle Väisälä Foundation of the Finnish Academy of Science and Letters, and the Waldemar von Frenckell and Magnus Ehrnrooth Foundations of the Finnish Society of Sciences and Letters. Part of this work was performed using an allocation of computing time from the Ohio Supercomputer Center.

References

- [1] S. Duane, A. D. Kennedy, B. J. Pendleton, D. Roweth, Phys. Lett. B **195**, 216 (1987);
- [2] S. A. Gottlieb, W. Liu, D. Toussaint, R. L. Renken, R. L. Sugar, Phys. Rev. D **35**, 2531 (1987).
- [3] M. R. Hestenes, E. Stiefel, J. Res. Nat. Bureau of Standards, **49**, 409 (1952).
- [4] C. Lanczos, J. Res. Nat. Bureau of Standards, **49**, 33 (1952); R. Fletcher, “Conjugate gradient methods for indefinite systems”; G. A. Watson (Ed.), *Numerical Analysis*, Lecture Notes in Mathematics, Vol. 506, (Springer, Berlin-Heidelberg, 1976).
- [5] H. A. van der Vorst, SIAM (Soc. Ind. Appl. Math.) J. Sci. Stat. Comput. **13**(2), 631 (1992).
- [6] C. G. Broyden, M. T. Vespucchi, *Krylov solvers for linear algebraic systems*, (Elsevier, Amsterdam, 2004).
- [7] Y. Saad, *Iterative methods for sparse linear systems*, (SIAM, Philadelphia, 2003).
- [8] T. DeGrand and C. DeTar, *Lattice Methods for Quantum Chromodynamics*, (World Scientific: Singapore, 2007); K. Orginos, J. Phys.: Conf. Ser. **46**, 132 (2006); T. A. DeGrand, *The presence and future of lattice QCD* in Computational Methods in Field Theory, edited by H. Gausterer and C.B. Lang, Lecture Notes in Physics 409 (Springer, 1992) 159; U. Wolff, *High precision simulations with fast algorithms* in Computational Methods in Field Theory, edited by H. Gausterer and C.B. Lang, Lecture Notes in Physics 409 (Springer, 1992) 127.
- [9] *Quantum Simulations of Condensed Matter Phenomena*, edited by J. D. Doll and J. E. Gubernatis (World Scientific, Singapore, 1990); *Quantum Monte Carlo Methods in Condensed Matter Physics*, edited by M. Suzuki (World Scientific, Singapore, 1993); W. von der Linden, Phys. Rep. **220**, 53 (1992).
- [10] E. Epelbaum, H. Krebs, D. Lee, Ulf-G. Meißner, Eur. Phys. J. A **40**, 199 (2009); Phys. Rev. Lett. **104**, 142501 (2010).
- [11] R. Blankenbecler, D. J. Scalapino, R. L. Sugar, Phys. Rev. D **24**, 2278 (1981).
- [12] T. Abe, R. Seki, Phys. Rev. C **79**, 054002 (2009); *ibid.* 054003 (2009).
- [13] H. F. Trotter, Proc. Am. Math. Soc. **10**, 545 (1959); M. Suzuki, Commun. Math. Phys. **51**, 183 (1976); Phys. Lett. A **146**, 319 (1990).
- [14] R. L. Stratonovich, Sov. Phys. Dokl. **2**, 416 (1958); J. Hubbard, Phys. Rev. Lett. **3**, 77 (1959).
- [15] D. Lee, Phys. Rev. C **78**, 024001 (2008); Prog. Part. Nucl. Phys. **63**, 117, (2009).
- [16] S. M. A. Rombouts, K. Heyde, N. Jachowicz, Phys. Rev. Lett. **82**, 4155 (1999).
- [17] J. Hirsch, Phys. Rev. B **28**, 4059(R) (1983).
- [18] R. T. Scalettar, D. J. Scalapino, R. L. Sugar, D. Toussaint, Phys. Rev. B **36**, 8632 (1987).
- [19] B. Bunk, S. Elser, R. Frezzotti, K. Jansen, Comput. Phys. Commun. **118**, 95 (1999).
- [20] R. Frezzotti, K. Jansen, Phys. Lett. B **402**, 328 (1997); Nucl. Phys. Proc. Suppl. **63**, 943 (1998); R. Frezzotti, K. Jansen, Nucl. Phys. B **555**, 395 (1999); *ibid.* **555**, 432 (1999).
- [21] A. van Wijngaarden, Indag. Math., **15**, 522 (1953).
- [22] H. A. Lauwerier, SIAM (Soc. Ind. Appl. Math.) J. Math. Anal. **6**, 96 (1975).
- [23] W. H. Press *et al.*, *Numerical Recipes in FORTRAN*, (2nd Ed., Cambridge University Press, Cambridge, England, 1992).